EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S131	1	"6542921".pn.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2006/12/21 10:13
S132	2	"6714958"	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2006/12/21 10:57
S133	1	"20040162706"	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2006/12/21 10:57
S134	608	lindholm.in.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:49
S135	3	(deadlock (dead same process)) and synchronized near2 object$1 same (wait$4 idl$3 sle$4 suspend$2)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:51
S136	2033	(deadlock (dead same process)) same (wait$4 idl$3 sle$4 suspend$2)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:52
S137	0	S134 and S136	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:51
S138	5244	deadlock	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:51
S139	0	S134 and S138	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:51
S140	2263	718/104,106,107.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:52
S141	689	710/200.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:52

EAST Search History

S142	81	(deadlock (dead same process)) same (wait$4 idl$3 sle$4 suspend$2) same time$out	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:58
S143	6	(S140 S141) and S142	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:53
S144	137	thread$1 and (deadlock (dead same process)) and event$1 same (awake woken wake$4)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 12:59
S145	4	(S140 S141) and S144	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:04
S146	16162	(process thread) same dead	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:05
S147	49	(S140 S141) and S146	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:14
S148	8	dead near lock same thread	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:16
S149	4	dead near lock and thread same (woke$4 wake$4 awake$3)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:49
S151	140	thread same (woke$4 wake$4 awake$3) same lock$1	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 07:43
S152	0	thread same (woke$4 wake$4 awake$3) same dead near lock	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:53
S153	2263	718/104,106,107.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:50
S154	689	710/200.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:50

EAST Search History

S156	44	S151 and (S153 S154)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/02 13:51
S157	4	thread same (woke$4 wake$4 awake$3) same dead	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 07:39
S158	2263	718/104,106,107.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 07:37
S159	689	710/200.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 07:37
S160	23	thread same (woke$4 wake$4 awake$3) same lock$1 and dead	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 10:48
S161	2	deadlock same (caus$4 occur$4) and (awoke waken)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 10:52
S162	51	deadlock$1 same (caus$4 occur$4) and (awoke waken awake)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:08
S163	52	deadlock$1 same (caus$4 occur$4 creat$3) and (awoke waken awake)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:09
S164	40	deadlock$1 same (creat$3) and (awoke waken awake)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:48
S165	1447	deadlock$1 same (idle$2 sleep wait$3)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:10
S166	926	deadlock$1 same (occur$4 creat$3 caus$4) same (idle$2 sleep wait$3)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:11
S167	926	deadlock$1 same (occur$4 creat$3 caus$4) same (idle$2 sleep slept wait$3)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:11

EAST Search History

S168	2263	718/104,106,107.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:11
S169	689	710/200.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:11
S170	97	S167 and (S168 S169)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:40
S171	17	dead$lock$1 same circular near wait	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:40
S172	282	deadlock$1 same (creat$3) and (awoke waken awake time$out)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:48
S173	709	deadlock$1 same (occur$3 happen$1 caus$3 creat$3) and (awoke waken awake time$out)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:48
S174	149	deadlock$1 same (occur$3 happen$1 caus$3 creat$3) same (awoke waken awake time$out)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:48
S175	18	(S168 S169) and S174	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/03 11:49
S176	3	latent near2 deadlock	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 10:16
S177	86	(li near zhen).xa.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 10:14
S178	0	S177 and S176	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 10:14
S179	54	latent and unsynchron$6	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 10:15

EAST Search History

S180	1	S177 and S179	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 10:15
S181	3	latent near2 deadlock$1	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 11:06
S183	2407	(caus$3 prevent$3 detect$3) near5 deadlock$1	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 11:06
S184	266	thread$1 near5 (woken awaken wake$3) same (acquir$4 request$3)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 11:08
S185	9	S183 and S184	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 12:16
S186	1	"6714958".pn.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2007/01/07 12:16

Sign in

Google Web Images Video News Maps more »
cause detect latent deadlocks Search Advanced Search Preferences

Web Results **1 - 10** of about **38,100** for **cause detect latent deadlocks**. (**0.27** seconds)

Detecting and causing **latent deadlocks** in multi-threaded programs ...
The predisposed execution of multi-threaded programs is modified to **cause** and **detect latent deadlocks**. When a thread attempts to.
www.freepatentsonline.com/6714958.html - 45k - Cached - Similar pages

[PDF] Communication **Deadlock Detection** of Inter-organizational Workflow ...
File Format: PDF/Adobe Acrobat
The **deadlock** problem may **cause** a long lasting workflow process to. S. Bhalla (Ed.): DNIS 2002, ... To explain how to statically **detect latent deadlock** in a ...
www.springerlink.com/index/JPUY3PWTDGALC6T3.pdf - Similar pages

[PDF] ReStore: Symptom Based Soft Error **Detection** in Microprocessors
File Format: PDF/Adobe Acrobat - View as HTML
back and **detection**. Thus, the **deadlock**, exception, and cfv. categories represent faults that are covered by ReStore while. faults in the sdc and **latent** ...
www.crhc.uiuc.edu/ACS/pub/dsn05.pdf - Similar pages

timed calculus of cryptographic communication
By using reachability tree analysis method to analyze the safety properties of the integrated model, some **latent deadlocks** can be detected. ...
www.iit.cnr.it/FAST2006/accepted.txt - 19k - Cached - Similar pages

docs.sun.com: Writing Device Drivers
Such values can be malignant, in that these values can **cause** a kernel panic if ... To avoid this situation, a device driver must **detect latent** faults and ...
docs.sun.com/app/docs/doc/816-4854/6mb1o3ahq?a=view - 39k - Cached - Similar pages

> docs.sun.com: Writing Device Drivers
> Such values can be malignant, meaning they can **cause** a kernel panic if ... To avoid this, a device driver must **detect latent** faults and report them in the ...
> docs.sun.com/app/docs/doc/806-5222/6je8fjvh0?a=view - 38k - Cached - Similar pages
> [More results from docs.sun.com]

Embedded.com - Where testing fails
The macro can be written to ignore cycles between resources that do not **cause** a **deadlock**. In the example shown in Figure 5, there are numerous cycles ...
www.embedded.com/showArticle.jhtml?articleID=12800625 - 104k - Cached - Similar pages

ReStore: Symptom-Based Soft Error **Detection** in Microprocessors
We are only interested in failures and only 8 percent of all trials (those that fall into the **deadlock**, exception, cfv, sdc, and **latent** categories) are ...
doi.ieeecomputersociety.org/10.1109/TDSC.2006.40 - Similar pages

[PDF] ReStore: Symptom Based Soft Error **Detection** in Microprocessors
File Format: PDF/Adobe Acrobat - View as HTML
can **cause** incorrect behavior. Since the error does not ... **detection** (baseline), with ReStore (ReStore, using **deadlock**,. exception, store miss symptoms and ...
www.cs.binghamton.edu/~oguz/war2005/papers/war_final1.pdf - Similar pages

[PDF] Problem Diagnosis in Large-Scale Computing Environments

File Format: PDF/Adobe Acrobat - View as HTML
locate the **cause** of a problem as follows. First, the. analyst injects the agent code into a running ... approach can **detect latent** faults that occur before ...
sc06.supercomputing.org/schedule/pdf/pap136.pdf - Similar pages

Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**

Get organized for the new year with Google Desktop.

cause detect latent deadlocks Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google

©2007 Google

Sign in

Google Web Images Video News Maps more »
cause latent deadlocks awake thread Search Advanced Search Preferences

Web Results **41** - **50** of about **15,500** for **cause latent deadlocks awake thread**. (**0.12** seconds)

full changelog
X mid-layer **deadlock** due to SCSI malloc fragmentation For pre-2.5. ... The **cause** of the slowdown is a 'swarm effect': if lots of **threads** are blocked on a ...
www.kernel.org/pub/linux/kernel/v2.5/ChangeLog-2.5.70 - 273k - Cached - Similar pages

Race control suspension - Patent 4532584
If a task is invoked by a processor with no **thread** of control, ... to insure that at least one CASE condition will always occur in order to avoid **deadlocks**. ...
www.freepatentsonline.com/4532584.html - 93k - Cached - Similar pages

[PDF] PubTeX output 2003.10.07:1003
File Format: PDF/Adobe Acrobat - View as HTML
Corrects a potential **deadlock** in the hardware configuration subsystem. ... Fixes a rare problem that **causes thread** blocking when waiting for memory. ...
h30097.www3.hp.com/docs/patch/51B/bl24/ReleaseNotes.pdf - Similar pages

Thinking in C++ 2nd ed Volume 2 rev 19
In order for a **thread** to **wake** up from a wait(), it must first reacquire the mutex that it ... Because all these conditions must be met to **cause deadlock**, ...
www.lsw.uni-heidelberg.de/users/sbrinkma/TICPP2/TicV2_III.html - 732k - Cached - Similar pages

Anonymous Weapons In Space Dialogue
Our **latent** colonialism strove with our longing for American unity. Our desire for peace strove ... It's now 4AM and I am definitely not fully **awake** anymore. ...
www.geocities.com/alienaxioms/space.html - 132k - Cached - Similar pages

[PS] RAIDframe: A Rapid Prototyping Tool for RAID Systems
File Format: Adobe PostScript - View as Text
The signal operation attempts to **wake** at most one **thread**, ... able to properly schedule its workload (and may **deadlock**). In its current release, ...
www.pdl.cmu.edu/PDL-FTP/RAID/CMU-CS-97-142.ps - Similar pages

Older, but no wiser
I have no idea what their **cause** was, whether it could be said by some ... we have reached a stage of **deadlock** in which we can think of nothing else to do. ...
olderandgrowing.blogspot.com/2003_11_01_olderandgrowing_archive.html - 101k - Cached - Similar pages

See also: My Computer Architecture Notes from Comps 2004 http ...
poison bits on result registers that **cause** exception when bad ... of error "in the service" alpha particle fault that **causes latent** error on read, ...
carlstrom.com/stanford/quals/arch.txt - 259k - Cached - Similar pages

[PDF] Software Construction and Analysis Tools for Future Space Missions
File Format: PDF/Adobe Acrobat - View as HTML
1) Software failures are **latent** design errors, and hence are very different from ... data race conditions, many of which could have led to **deadlock**. ...
www-etaps.imag.fr/Invited/Lowry-tacas.pdf - Similar pages

Spirit of Vatican II: 5. The Three Poisons
Any **cause**, however noble, is analyzed as a potential source of delusion and bondage. ...
Each of the Three Poisons brings the others in its **wake**. ...
josephsoleary.typepad.com/my_weblog/2006/10/5_the_three_poi.html - 119k -
Cached - Similar pages

Result Page: **Previous** 1 2 3 4 5 6 7 8 9 10 11 12 13 14 **Next**

Get organized for the new year with Google Desktop.

cause latent deadlocks awake threa Search

Search within results | Language Tools | Search Tips

Google Home - Advertising Programs - Business Solutions - About Google

©2007 Google

PORTAL USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: The ACM Digital Library The Guide

cause detect latent deadlocks

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **cause detect latent deadlocks** Found **30,756** of **193,448**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown Relevance scale

1 Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research**

Publisher: IBM Press

Full text available: pdf(4.21 MB) Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display repeated occurrences of non-trivial commun ...

2 Improving the reliability of commodity operating systems

Michael M. Swift, Brian N. Bershad, Henry M. Levy

February 2005 **ACM Transactions on Computer Systems (TOCS)**, Volume 23 Issue 1

Publisher: ACM Press

Full text available: pdf(459.98 KB) Additional Information: full citation, abstract, references, index terms

Despite decades of research in extensible operating system technology, extensions such as device drivers remain a significant cause of system failures. In Windows XP, for example, drivers account for 85&percent; of recently reported failures.This article describes Nooks, a *reliability subsystem* that seeks to greatly enhance operating system (OS) reliability by isolating the OS from driver failures. The Nooks approach is practical: rather than guaranteeing complete fault tolerance through ...

Keywords: I/O, Recovery, device drivers, protection, virtual memory

3 Making operating systems more robust: Improving the reliability of commodity operating systems

Michael M. Swift, Brian N. Bershad, Henry M. Levy

October 2003 **Proceedings of the nineteenth ACM symposium on Operating systems principles**

Publisher: ACM Press

Full text available: pdf(262.78 KB) Additional Information: full citation, abstract, references, citings, index terms

Despite decades of research in extensible operating system technology, extensions such as device drivers remain a significant cause of system failures. In Windows XP, for example, drivers account for 85% of recently reported failures. This paper describes Nooks, a *reliability subsystem* that seeks to greatly enhance OS reliability by isolating the OS from driver failures. The Nooks approach is practical: rather than guaranteeing complete fault tolerance through a new (and incompatible) OS ...

Keywords: I/O, device drivers, protection, recovery, virtual memory

4 Verification: Automated, scalable debugging of MPI programs with Intel® Message Checker

Jayant DeSouza, Bob Kuhn, Bronis R. de Supinski

May 2005 **Proceedings of the second international workshop on Software engineering for high performance computing system applications SE-HPCS '05**

Publisher: ACM Press

Full text available: pdf(239.87 KB) Additional Information: full citation, abstract, references, index terms

The trend towards many-core multi-processor systems and clusters will make systems with tens and hundreds of processors more widely available. Current manual debugging techniques do not scale well to such large systems. Advanced automated debugging tools are needed for standard programming models based on commodity computing, such as threads and MPI. We surveyed MPI users to identify the kinds of MPI errors that they encounter, and classify the errors into several types. We describe how automate ...

Keywords: MPI, MPI implementation validation, MPI standard checking, automated and scalable debugging, correctness/confidence tools, protocol verification

5 Automatic detection and repair of errors in data structures

Brian Demsky, Martin Rinard

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications OOPSLA '03**, Volume 38 Issue 11

Publisher: ACM Press

Full text available: pdf(340.56 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a system that accepts a specification of key data structure consistency constraints, then dynamically detects and repairs violations of these constraints, enabling the program to continue to execute productively even in the face of otherwise crippling errors. Our experience using our system indicates that the specifications are relatively easy to develop once one understands the data structures. Furthermore, for our set of benchmark applications, our system can effectively repair inco ...

Keywords: data structure invariants, data structure repair

6 Using meta-level compilation to check FLASH protocol code

Andy Chou, Benjamin Chelf, Dawson Engler, Mark Heinrich

November 2000 **ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , Proceedings of the ninth international conference on Architectural support for programming languages and operating systems ASPLOS-IX**, Volume 34 , 28 Issue 5 , 5

Publisher: ACM Press

Additional Information: full citation, abstract, references, citings, index

Full text available: pdf(297.02 KB) terms

Building systems such as OS kernels and embedded software is difficult. An important source of this difficulty is the numerous rules they must obey: interrupts cannot be disabled for ~too long," global variables must be protected by locks, user pointers passed to OS code must be checked for safety before use, etc. A single violation can crash the system, yet typically these invariants are unchecked, existing only on paper or in the implementor's mind.This paper is a case study in how system impl ...

7 Using meta-level compilation to check FLASH protocol code

Andy Chou, Benjamin Chelf, Dawson Engler, Mark Heinrich

November 2000 **ACM SIGPLAN Notices**, Volume 35 Issue 11

Publisher: ACM Press

Full text available: pdf(1.39 MB) Additional Information: full citation, abstract, references, index terms

Building systems such as OS kernels and embedded software is difficult. An important source of this difficulty is the numerous rules they must obey: interrupts cannot be disabled for "too long," global variables must be protected by locks, user pointers passed to OS code must be checked for safety before use, etc. A single violation can crash the system, yet typically these invariants are unchecked, existing only on paper or in the implementor's mind.This paper is a case study in how system impl ...

8 Rx: treating bugs as allergies---a safe method to survive software failures

Feng Qin, Joseph Tucek, Jagadeesan Sundaresan, Yuanyuan Zhou

October 2005 **ACM SIGOPS Operating Systems Review , Proceedings of the twentieth ACM symposium on Operating systems principles SOSP '05**, Volume 39 Issue 5

Publisher: ACM Press

Full text available: pdf(245.29 KB) Additional Information: full citation, abstract, references, index terms

Many applications demand availability. Unfortunately, software failures greatly reduce system availability. Prior work on surviving software failures suffers from one or more of the following limitations: Required application restructuring, inability to address deterministic software bugs, unsafe speculation on program execution, and long recovery time.This paper proposes an innovative *safe* technique, called Rx, which can quickly recover programs from many types of software bugs, both det ...

Keywords: availability, bug, reliability, software failure

9 OOPSLA onward!: Finding bugs is easy

David Hovemeyer, William Pugh

December 2004 **ACM SIGPLAN Notices**, Volume 39 Issue 12

Publisher: ACM Press

Full text available: pdf(506.92 KB) Additional Information: full citation, abstract, references

Many techniques have been developed over the years to automatically find bugs in software. Often, these techniques rely on formal methods and sophisticated program analysis. While these techniques are valuable, they can be difficult to apply, and they aren't always effective in finding real bugs.*Bug patterns* are code idioms that are often errors. We have implemented automatic detectors for a variety of bug patterns found in Java programs. In this paper, we describe how we have used bug pa ...

10 Formal modeling and analysis of the HLA component integration standard

Robert J. Allen, David Garlan, James Ivers

November 1998 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 6th ACM SIGSOFT international symposium on Foundations of software**

engineering SIGSOFT '98/FSE-6, Volume 23 Issue 6
Publisher: ACM Press
Full text available: pdf(1.21 MB) Additional Information: full citation, abstract, references, citings, index terms

An increasingly important trend in the engineering of complex systems is the design of component integration standards. Such standards define rules of interaction and shared communication infrastructure that permit composition of systems out of independently-developed parts. A problem with these standards is that it is often difficult to understand exactly what they require and provide, and to analyze them in order to understand their deeper properties. In this paper we use our experience in mod ...

Keywords: component integration standards, component-based software, formal specification, protocol families, software architecture

11 Mondrix: memory isolation for linux using mondriaan memory protection
Emmett Witchel, Junghwan Rhee, Krste Asanović
October 2005 **ACM SIGOPS Operating Systems Review , Proceedings of the twentieth ACM symposium on Operating systems principles SOSP '05**, Volume 39 Issue 5
Publisher: ACM Press
Full text available: pdf(332.09 KB) Additional Information: full citation, abstract, references, index terms

This paper presents the design and an evaluation of Mondrix, a version of the Linux kernel with Mondriaan Memory Protection (MMP). MMP is a combination of hardware and software that provides efficient fine-grained memory protection between multiple protection domains sharing a linear address space. Mondrix uses MMP to enforce isolation between kernel modules which helps detect bugs, limits their damage, and improves kernel robustness and maintainability. During development, MMP exposed two kerne ...

Keywords: fine-grained memory protection

12 TSOtool: A Program for Verifying Memory Systems Using the Memory Consistency Model
Sudheendra Hangal, Durgam Vahia, Chaiyasit Manovit, Juin-Yeu Joseph Lu
March 2004 **ACM SIGARCH Computer Architecture News , Proceedings of the 31st annual international symposium on Computer architecture ISCA '04**, Volume 32 Issue 2
Publisher: IEEE Computer Society, ACM Press
Full text available: pdf(206.30 KB) Additional Information: full citation, abstract, citings

In this paper, we describe TSOtool, a program to check thebehavior of the memory subsystem in a shared memorymultiprocessor. TSOtool runs pseudo-randomly generatedprograms with data races on a system compliant with theTotal Store Order (TSO) memory consistency model; it thenchecks the results of the program against the formal TSOspecification. Such analysis can expose subtle memory errorslike data corruption, atomicity violation and illegalinstruction ordering.While verifying TSO compliance comp ...

Keywords: Memory consistency models, Multiprocessor verification, Sequential Consistency, Total Store Order

13 Making operating systems more robust: RacerX: effective, static detection of race conditions and deadlocks
Dawson Engler, Ken Ashcraft

October 2003 **Proceedings of the nineteenth ACM symposium on Operating systems principles**
Publisher: ACM Press
Full text available: pdf(310.63 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper describes RacerX, a static tool that uses flow-sensitive, interprocedural analysis to detect both race conditions and deadlocks. It is explicitly designed to find errors in large, complex multithreaded systems. It aggressively infers checking information such as which locks protect which operations, which code contexts are multithreaded, and which shared accesses are dangerous. It tracks a set of code features which it uses to sort errors both from most to least severe. It uses novel ...

Keywords: deadlock detection, program checking, race detection

14 A VISUAL ENVIRONMENT FOR DISTRIBUTED SIMULATION SYSTEMS
James H. Graham, Adel S. Elmaghraby, Irfan Karachiwala, Hussam Soliman
January 1996 **ACM SIGSIM Simulation Digest**, Volume 25 Issue 3
Publisher: ACM Press
Full text available: pdf(919.47 KB) Additional Information: full citation, abstract, references

Parallel and Distributed Simulation (PADS) algorithms are typically categorized to belong to one of two categories. They are either conservative or optimistic with respect to the method of handling causality. Conservative systems strictly preserve causality, while optimistic systems detect and correct causality errors when they occur. *Time Warp* is the basis of optimistic algorithms where rolling back the simulation clock allows the simulation to correct for errors. The Global Virtual Time ...

15 Performance modeling and analysis: From UML sequence diagrams and statecharts to analysable petri net models
Simona Bernardi, Susanna Donatelli, José Merseguer
July 2002 **Proceedings of the 3rd international workshop on Software and performance WOSP '02**
Publisher: ACM Press
Full text available: pdf(246.31 KB) Additional Information: full citation, abstract, references, citings

In this paper we study the use of Sequence Diagrams and Statecharts of the Unified Modeling Language (UML) for the validation and the performance evaluation of systems. We assume that the system is specified as a set of Statecharts and that Sequence Diagrams are used to represent "executions of interest". UML lacks a formal semantics and hence it is not possible to apply, directly, mathematical techniques on UML models for system validation. To reach this goal we propose an automatic translation ...

Keywords: GSPN, UML, compositionality, modeling

16 Guidance for the use of the Ada programming language in high integrity systems
B. A. Wichmann
July 1998 **ACM SIGAda Ada Letters**, Volume XVIII Issue 4
Publisher: ACM Press
Full text available: pdf(2.93 MB) Additional Information: full citation, abstract, citings, index terms

This paper is the current result of a study by the ISO HRG Rapporteur group which is being circulated for comment. Many people have contributed to this, but those who have either attended two recent meetings of group or have made substantial e-mail comments are: Praful V Bhansali (Boeing, USA), Alan Burns (University of York, UK), Bernard

Carre' (Praxis Critical Systems, UK), Dan Craigen (ORA, Canada), Nick Johnson MoD, UK), Stephen Michell (Canada), Gilles Motet (DGEI/INSA, France), George Roma ...

17 Research papers: test & analysis IV: Perracotta: mining temporal API rules from imperfect traces

Jinlin Yang, David Evans, Deepali Bhardwaj, Thirumalesh Bhat, Manuvir Das

May 2006 **Proceeding of the 28th international conference on Software engineering ICSE '06**

Publisher: ACM Press

Full text available: pdf(174.43 KB) Additional Information: full citation, abstract, references, index terms

Dynamic inference techniques have been demonstrated to provide useful support for various software engineering tasks including bug finding, test suite evaluation and improvement, and specification generation. To date, however, dynamic inference has only been used effectively on small programs under controlled conditions. In this paper, we identify reasons why scaling dynamic inference techniques has proven difficult, and introduce solutions that enable a dynamic inference technique to scale to l ...

Keywords: dynamic analysis, specification inference, temporal properties

18 Robust service: Rewind, repair, replay: three R's to dependability

Aaron B. Brown, David A. Patterson

July 2002 **Proceedings of the 10th workshop on ACM SIGOPS European workshop: beyond the PC EW10**

Publisher: ACM Press

Full text available: pdf(146.14 KB) Additional Information: full citation, abstract, references

Motivated by the growth of web and infrastructure services and their susceptibility to human operator-related failures, we introduce *system-level undo* as a recovery mechanism designed to improve service dependability. Undo enables system operators to recover from their inevitable mistakes and furthermore enables *retroactive repair* of problems that were not fixed quickly enough to prevent detrimental effects. We present the "three R's", a model of undo that matches the needs of huma ...

19 Special session on memory wall: Fighting the memory wall with assisted execution

Michel Dubois

April 2004 **Proceedings of the 1st conference on Computing frontiers**

Publisher: ACM Press

Full text available: pdf(231.18 KB) Additional Information: full citation, abstract, references, index terms

Assisted execution is a form of simultaneous multithreading in which a set of auxiliary "assistant" threads, called *nanothreads*, is attached to each thread of an application. Nanothreads are lightweight threads which run on the same processor as the main (application) thread and help execute the main thread as fast as possible. Nanothreads exploit resources that are idled in the processor because of hazards due to program dependencies and memory access delays.Assisted execution has the po ...

Keywords: cache memories, latency tolerance, prefetching, simultaneous multithreading, superscalar processors

20 Flow-sensitive type qualifiers

Jeffrey S. Foster, Tachio Terauchi, Alex Aiken

May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation PLDI '02**, Volume 37 Issue 5

Publisher: ACM Press

Full text available: pdf(298.28 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a system for extending standard type systems with flow-sensitive type qualifiers. Users annotate their programs with type qualifiers, and inference checks that the annotations are correct. In our system only the type qualifiers are modeled flow-sensitively---the underlying standard types are unchanged, which allows us to obtain an efficient constraint-based inference algorithm that integrates flow-insensitive alias analysis, effect inference, and ideas from linear type systems to supp ...

Keywords: alias analysis, constraints, effect inference, flow-sensitivity, linux kernel, locking, restrict, type qualifiers, types

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player